AF
2700

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Tran**

Serial No.: **09/550,181**

Filed: **April 14, 2000**

For: **Method and Apparatus for Maintaining State Information for Web Pages Using a Directory Server**

Group Art Unit: **2155**

Examiner: **Wang, Liang Che A.**

Attorney Docket No.: **AUS000078US1**

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

Certificate of Mailing Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being deposited with the United States Postal Service as First Class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on April 20, 2004.

By: Michele Morrow
Michele Morrow

TRANSMITTAL DOCUMENT

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:
ENCLOSED HEREWITH:

- Appellant's Brief (in triplicate) (37 C.F.R. 1.192); and
- Our return postcard.

A fee of $330.00 is required for filing an Appellant's Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

Respectfully submitted,

Duke W. Yee
Registration No. 34,285
YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 367-2001
ATTORNEY FOR APPLICANT

Docket No. AUS000078US1 *PATENT*

#8
LOO
5-5-04

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Tran**

Serial No. **09/550,181**

Filed: **April 14, 2000**

For: **Method and Apparatus for Maintaining State Information for Web Pages Using a Directory Server**

Group Art Unit: **2155**

Examiner: **Wang, Liang Che A.**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

ATTENTION: Board of Patent Appeals and Interferences

Certificate of Mailing Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being deposited with the United States Postal Service as First Class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on April 20, 2004.

By: Michele Morrow
Michele Morrow

04/27/2004 MBERHE 00000114 090447 09550181
01 FC:1402 330.00 DA

APPELLANT'S BRIEF (37 C.F.R. 1.192)

This brief is in furtherance of the Notice of Appeal, filed in this case on February 20, 2004.

The fees required under § 1.17(c), and any required petition for extension of time for filing this brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

This brief is transmitted in triplicate. (37 C.F.R. 1.192(a))

04/26/2004 MBERHE 00000019 090447 09550181
01 FC:1401 330.00 DA

Adjustment date: 04/27/2004 MBERHE
04/26/2004 MBERHE 00000019 090447 09550181
01 FC:1401 330.00 CR

REAL PARTIES IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines, Inc.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-11, 13-19 and 21-40

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: NONE
2. Claims withdrawn from consideration but not canceled: NONE
3. Claims pending: 1-11, 13-19 and 21-40
4. Claims allowed: NONE
5. Claims rejected: 1-11, 13-19 and 21-40

C. CLAIMS ON APPEAL

The claims on appeal are: 1-11, 13-19 and 21-40

STATUS OF AMENDMENTS

No amendments to the claims after final rejection have been made.

SUMMARY OF INVENTION

The present invention provides a method, apparatus and system for maintaining and storing Web page state information for every Web page previously visited by a user in a directory server. When the user requests a Web page, a request is sent through a background application to the directory server. If the Web page information is stored in the directory server, the user input is inserted into the Web page before downloading to the user's device. If the Web page information is not stored in the directory server, a new Web page entry is created in the directory server to store the user's input into the Web page.

ISSUES

The issues on appeal are (1) whether claims 1-7, 9, 11, 13-18, 21-27, 29-33 and 35-40 are unpatentable over Larsen (U.S. Patent No. 6,088,700), (2) whether claims 8, 19, 28 and 34 are unpatentable over Larsen in view of Tomko (U.S. Patent No. 5,790,668), and (3) whether claims 10 and 36 are unpatentable over Larsen in view of Call (U.S. Patent No. 6,154,738).

GROUPING OF CLAIMS

The claims do not stand or fall together for the reasons set forth hereafter in Appellant's arguments. The claims stand or fall according to the following grouping of claims:

Group I: claims 1, 13, 21 and 37;

Group II: claims 2, 14, 22 and 30;

Group III: claims 3, 15, 23 and 31;

Group IV: claims 4, 16 and 24;

Group V: claims 5, 17 and 25;

Group VI: claims 6, 18 and 26;

Group VII: claims 7, 27 and 33;

Group VIII: claims 8, 19, 28 and 34;[1]

Group IX: claims 9 and 35;

Group X: claims 10 and 36;

Group XI: claim 11;

Group XII: claim 29;

Group XIII: claim 32; and

Group XIV: claims 38, 39 and 40;

[1] It is noted that claim 19 in the Response to Office Action filed on October 15, 2003 recited identical features as claim 18 by mistake. No amendment to claim 19 was intended as is clearly evident from the manner in which claim 19 appears in the October 15, 2003 Response, i.e. no additions or deletions with regard to the original language of claim 19 are shown. In addition, the Final Office Action rejects claim 19 as if it contains the same features recited in the originally filed claim 19. Therefore, Appellant respectfully submits that claim 19 was not amended and the Examiner did not regard claim 19 as having been amended. Accordingly, claim 19 as it appears in the Appendix of Claims of this Appeal Brief, recites the same features as the originally filed claim.

ARGUMENT

I. 35 U.S.C. § 103(a), Alleged Obviousness, Claims 1-7, 9, 11, 13-18, 21-27, 29-33 and 35-40

The Final Office Action rejects claims 1-7, 9, 11, 13-18, 21-27, 29-33 and 35-40 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Larsen et al.(U.S. Patent No. 6,088,700). This rejection is respectfully traversed.

With regard to independent claim 1, the Final Office Action states:

> Referring to claim 1, Larsen has taught a method for maintaining state information for web page (see title and abstract), comprising:
>
> receiving user input to a Web page via a Web browser at a client device (abstract lines 3-4, fill out data field... multiple forms....on the web browser of users...Col 2 lines 8-13, 30-33, Col 3 lines 28-30, Col 17 lines 14-16, the user input is used for future automatically form filling on the web page via a browser of users);
>
> sending an instruction to store user input and a Web identifier in a directory server (abstract lines 7-9, "the data process system retrieves tagged information previously entered and stored in the database"; abstract line 9, "sending instruction for storing is inherent since computer processing requires instruction to complete tasks, therefore the step of sending an instruction must be in Larsen in order to process the storing step;)
>
> storing the user input and a corresponding web page field identifier (abstract, Col 17 lines 25-45) in the directory server (Col 3 lines 12-14, database of Larsen is a directory server); and
>
> in response to receiving a user request, via the Web browser, for the Web page, sending a request to the directory server to retrieve the user input and corresponding Web page identifier (Col 3 line 59-Col 4 line 8, Abstract lines 7-13), wherein the user input and corresponding Web page field identifier are retrieved from the directory server (abstract, Col 2, lines 8-11, 20-38.)
>
> Although Larsen has not explicitly taught using a background application at the client device to send instruction and retrieve the user input. But, Larsen does teach the data processing system retrieves previously saved information and automatically inserts the data in a variety of forms (abstraction lines 7-13.) It is known that every processing system has instruction being processed in order to perform functions. And the instructions are viewed as background application since they are both just groups of computer codes that perform particular tasks. Without this background application, a processing system will not function, therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the teaching of Larsen such that to implemented his invention by using a background application.
>
> A person with ordinary skill in the art would have been motivated to make the modification to Larsen because background application is well known in the

art as computer codes that runs the processing system, and a person with ordinary skill in the art would have implemented Larsen's invention using various of well known type of application including plug-in and background applications to enhance the functionalities of Larsen's computer system.

Final Office Action dated November 20, 2003, pages 2-4.

Independent claim 1, which is representative of independent claims 13 and 21 with regard to similarly recited subject matter, reads as follows:

> 1. A method for maintaining state information for Web pages, comprising:
> receiving user input to a Web page via a Web browser at a client device;
> sending an instruction to store user input and a Web page field identifier in a directory server from a background application running on the client device;
> storing the user input and a corresponding Web page field identifier in the directory server; and
> in response to receiving a user request, via the Web browser, for the Web page, sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier, wherein the user input and corresponding Web page field identifier are retrieved from the directory server.
> (emphasis added)

Larsen teaches a system that automatically completes multiple forms for multiple companies or regulatory agencies by filling out information once on any form presented on the user's browser. With the Larsen system, a company must register its forms, which may then be populated with information entered by a user. The user in Larsen enters a form identifier and user input to be entered into the form. The data processing system retrieves tagged information previously entered by the user and stored in a database, and automatically inserts the data in similarly tagged uncompleted fields of any number and variety of registered forms (column 2, lines 24-35). Larsen uses a Mapping Application to identify the forms, their fields and the clients subscribing to each form, and then map that information into a relational database. Larsen then uses Web Form Filler that uses a web server that allows potential candidates or employees of a company to fill out the forms electronically (column 2, lines 8-12).

Larsen does not teach or suggest sending an instruction to store user input and a Web page field identifier in a directory server from a background application running on the client device or sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier. Larsen teaches at column 3, lines 61-67 that a Web user requests a form over the Global Information Network or

Local Network and enters information on his or her display. The information is then transferred over the network to a Form Parser, which is a network server. However, Larsen does not mention anything about an application that is running in the background on the Web user's device or any background application that sends instruction to store user input and a Web page field identifier or sends request to retrieve the user input and corresponding Web page field identifier.

With the background application of the presently claimed invention, a user thinks that he or she is visiting a Web page in a normal fashion, however, specific functions are performed in the background to retrieve user input and a Web page field identifier and populate the Web page without the user's awareness. Larsen does not teach such features. To the contrary, Larsen teaches that a user has to request a form and enter information on his or her display. The system of Larsen also does not perform specific functions, such as sending an instruction in the background to store user input and a Web page field identifier in the directory server without user's request.

The Final Office Action admits that Larsen does not explicitly teach using a background application at the client device to send instruction and retrieve the user input. However, the Final Office Action alleges that it is known that every processing system processes instructions to perform functions and the instructions are viewed as a background application, since they are only groups of computer codes that perform particular tasks. The Final Office Action further alleges that without this background application, a processing system will not function, and therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the teaching of Larsen in order to implement his invention by using a background application.

However, Appellant disagrees with the above allegations made in the Final Office Action. At column 3, lines 61-67, Larsen teaches that a Web User (12), company candidate or employee, requests a form over the Global information Network or Local Network and enters information on his or her display. The information then transferred over the network to the Form Parser (13), which is a network server. The Form Parser (13) submits the request for the form and inputted data to the Form Database (3). Larsen does not teach that any background application exists on the Web user's device, or any other application on the Web user's device, that sends and retrieves the user input and corresponding Web page field identifier. To the contrary, Larsen teaches that the Form Parser, which is implemented in a network server separate from the Web user's device, submits the request for the form, and input data from the Web user, to the Form Database.

In addition, the background application of the presently claimed invention is an application that runs in the background of a Web browser on a client device, such that the user is not aware of the operations being performed by the background application. The background application runs in parallel with the Web browser on the client device. Therefore, the client device will still be fully functional without the background application of the presently claimed invention, since other applications, such as a Web browser, may still be running on the client device. The background application of the presently claimed invention has nothing to do with whether a client device will function as a data processing system. The background application of the presently claimed invention has to do with sending an instruction to store user input and a Web page field identifier in a directory server, and retrieving user input and Web page field identifier from the directory server, as well as populating the Web page without a user's awareness when the user requests a Web page.

Therefore, it would not have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the teaching of Larsen to implement sending an instruction from a background application to store user input and a Web page field identifier in a directory server or sending a request from a background application to a directory server to retrieve user input and a corresponding Web page field identifier, because not only does Larsen fail to teach or suggest a background application running on the client device that performs these features, the allegations made in the Final Office Action are erroneous.

In the Response to Arguments section, the Final Office Action alleges that a person with ordinary skill in the art would know that there must be a background application running on the client device in order for the client device to send the user input to be stored in the remote database server, or the client device would not function. However, the Final Office Action fails to point out where in Larsen such an allegation is supported. The Examiner merely asserts that a person of ordinary skill in the art would know that a background application must be run on the client device in order for the client device to function. To the contrary of the Examiner's assertion, a person of ordinary skill in the art would know that even without the background application running on the client device, other applications, such as a Web browser, can still run on the client device and that the client device would still be functional.

However, even if it were true that a background application were necessary to send

information to a server, there is nothing in Larsen that teaches a background application that specifically sends an instruction to store user input and a Web page field identifier in a directory server or sends a request to retrieve stored user input and a corresponding Web page field identifier.

Furthermore, the Final Office Action alleges that a person with ordinary skill in the art would have been motivated to make the modification to Larsen because the background application is well known in the art as computer codes that run the processing system, and a person with ordinary skill in the art would have implemented Larsen's invention using various of well known type of applications including plug-in and background application to enhance the functionalities of Larsen's computer system. Appellant respectfully disagrees. The background application of the presently claimed invention is not the same as well known computer codes that run on a data processing system, in that the background application of the presently claimed invention runs in parallel with a Web browser and performs specific functions in the background, such as sending an instruction to store user input and a Web page field identifier and sending a request to retrieve stored user input and corresponding Web page field identifier. To the contrary, ordinary computer codes are merely instructions running on a data processing system that are hidden from a user's view. Ordinary computer codes do not perform specific functions, such as sending instruction to a directory server to store user input and a Web page field identifier. Just because computer codes are running on a data processing system and are hidden from a user's view does not make them the same as a background application. Therefore, a person of ordinary skill in the art would not have been motivated to make the modification to Larsen since well known computer codes running on a data processing system are not the same as a background application and that they do not perform specific functions such as those mentioned in claim 1. Furthermore, there is no teaching or suggestion anywhere in Larsen about a background application or a plug-in that runs parallel to a Web browser on the Web user's device, let alone a background application that sends an instruction to store user input and Web page field identifier and sends a request to retrieve user input and Web page field identifier.

In addition, Larsen is only concerned with a network server, such as the Form Parser, that allows a user to complete multiple pre-filled forms for multiple companies or regulatory agencies. Larsen is not concerned with having a background application running in parallel with a Web browser on the Web user's device, i.e. client device, that sends instruction to store user

input and Web page field identifier in a directory server and sends request to retrieve user input and corresponding Web page field identifier from directory server. A person of ordinary skill in the art would not have been motivated to modify Larsen's teaching to include a background application to enhance functionalities of a Web user's device because Larsen is only interested in enhancing functionalities outside of the Web user's device to assist in completing multiple forms.

In view of the above, Appellant respectfully submits that Larsen does not teach or suggest the features recited in independent claim 1. The other independent claims 13 and 21 recite similar subject matter also not taught by Larsen. Thus, Appellant respectfully requests withdrawal of the rejection of claims 13 and 21.

In addition, Larsen does not teach the specific features of independent claim 29. Independent claim 29, recites:

> 29. A method, in a directory server, for maintaining state information for Web pages, comprising:
>
> receiving user input to a Web page from a background application running on a client device; and
>
> storing the user input and a corresponding Web page field identifier received from the background application running on the client device in the directory server, wherein the user input and corresponding Web page field identifier are downloaded from the directory server to the client device in response to a user request for a Web page.
>
> (emphasis added)

Larsen does not teach or suggest a method in a directory server that receives user input to a Web page from a background application running on a client device. Larsen only teaches, in column 3, lines 61-63, a system that receives a request from a user for a form over a Global Information Network or Local Network and enters information on his or her display. In the Abstract, Larsen teaches to use data stored in a database to automatically fill out data fields of the forms displayed on the Web browser. Larsen never suggests what the user's display actually is. Assuming that the display is a normal Web browser, as stated in the Abstract, Larsen does not teach or suggest receiving user input to a Web page from a background application on the client device that stores user input and Web page field identifier in the directory server, as recited in claim 29. As discussed in the arguments presented for claim 1, a background application is an application that runs in parallel with a Web browser and performs specific functions in the

background without a user's request. Larsen only teaches a display that allows a user to enter his or her information. Larsen does not teach a background application that runs in parallel with the display or a background application that stores user input and Web page field identifier in a directory server. Thus, Appellant submits that Larsen does not teach or suggest the specific features recited in independent claim 29. Accordingly, Appellant respectfully requests withdrawal of the rejection of claim 29 under 35 U.S.C. § 103(a).

With regard to independent claim 37, Larsen does not teach or suggest sending an instruction, from a background application running on the client, to the server to store user input to the Web page and the identifier associated with the user input. Larsen also does not teach or suggest that responsive to a subsequent request, via a Web browser for the Web page, sending a request from a background application running on the client to the server to retrieve the user input and the identifier from the server. As discussed in the arguments presented above for claims 1, 13 and 21, Larsen does not teach any background application running on the client that sends an instruction to store user input and identifier associated with the user input in a directory server or sends a request to retrieve user input and the identifier response to a subsequent request for a Web page. Larsen only teaches that a user may request a form over the network and enters information on his or her display and the information is then transferred over the network to a Form Parser, which is a network server. There is no mention of a background application running on the client that performs specific functions, such as those mentioned in claim 37. Since claim 37 recites similar features recited in claims 1, 13 and 21, Larsen also does not teach the features of claim 37. Accordingly, Appellant respectfully requests withdrawal of the rejection of claim 37 under 35 U.S.C. § 103(a).

At least by virtue of their dependency on claims 1, 13, 21, 29 and 37, respectively, Larsen does not teach or suggest the features set forth in dependent claims 2-7, 9, 11, 14-18, 22-27, 30-33, 35, 36 and 38-40. Accordingly, Appellant respectfully requests withdrawal of the rejection of claims 1-7, 9, 11, 13-18, 21-27, 29-33 and 35-40 under 35 U.S.C. § 103(a).

In addition, Larsen does not teach or suggest any of the specific features set forth in the dependent claims 2-7, 9, 11, 14-18, 22-27, 30-33, 35, 36 and 38-40. For example, with regard to dependent claim 2, which is representative of claims 14, 22 and 30 with regard to similarly recited subject matter, Larsen does not teach or suggest that the user input and Web page field identifier are specific to a particular Web page. The Final Office Action alleges that Larsen teaches this feature

at column 17, lines 25-29, which reads as follows:

> said registration means begin capable of assigning said form template a unique registration such that said form template becomes a uniquely registered form template.

In the above section, Larsen teaches assigning a registration to a form template that is unique from other registered form templates. Thus, Larsen only teaches a registration that is specific to a form template. Larsen does not teach or suggest that both the user input and the Web page field identifier are specific to a particular Web page. Therefore, Larsen does not teach the features of claims 2, 14, 22 and 30.

With regard to dependent claim 3, which is representative of claims 15, 23 and 31 with regard to similarly recited subject matter, Larsen does not teach or suggest that the user input and Web page field identifier are common to a plurality of Web pages. The Final Office Action alleges that Larsen teaches this feature at column 17, lines 34-38, which reads as follows:

> said mapping means being capable of sorting said uniquely registered form identifying information into common group information.

In the above section, Larsen teaches mapping means that include sorting identifying information that identifies a unique registered form into common group information, which implies that the identifying information may be sorted based on the common group information. However, Larsen does not teach that the user input and Web page field identifier are common to a plurality of Web pages. Larsen does not teach that the common group information includes a plurality of Web pages. Larsen also does not teach that the identifying information includes user input and Web page field identifier. Larsen only teaches that the identifying information identifies each uniquely registered form. In other words, forms may be grouped based on these identifiers. Larsen does not teach or suggest that user input and a Web page field identifier are common to a plurality of Web pages, as recited in claims 3, 15, 23 and 31.

With regard to dependent claim 4, which is representative of claims 16 and 24 with regard to similarly recited subject matter, Larsen does not teach or suggest matching the Web page field identifier to an entry field identifier located in the Web page or inserting the user input into a field associated with the entry field identifier. The Final Office Action alleges that Larsen teaches these features at column 17, lines 46-56 and in the Abstract, lines 7-11, which reads as follows:

> said matching means being capable of matching said uniquely identified form template data

fields to said common group information such as to allow matches to be verified by said entity using said data processing system,

said uniquely identified form template data fields which are verifiably matched to said common group information becoming common form fields stored in said database.

The data processing system retrieves tagged information previously entered and stored in a database, and automatically inserts the data in similarly tagged uncompleted fields of any number and variety of forms.

In the first section, Larsen teaches that data fields in a uniquely identified form template are matched to common group information in order to allow an entity to verify the matches. In addition, data fields in a uniquely identified form template that are verified become common form fields that are stored in the database. In the second section, Larsen teaches that tagged information previously entered and stored in the database is retrieved and inserted in similarly tagged uncompleted fields of any form. However, neither of the above sections teaches nor suggests that the Web page field identifier is matched to an entry identifier located in the Web page. In Larsen, the data fields are matched to common group information for verification by an entity. The data fields are not matched to an entry identifier located in a Web page. Nowhere in the reference does Larsen teach or suggest an entry identifier that is located in a Web page. In addition, Larsen teaches that data is inserted in similarly tagged uncompleted fields of any form. Thus, Larsen only inserts data to the uncompleted field if the tag of the uncompleted field is similar to the tag that is retrieved from the database. Larsen does not teach that the data is inserted into a field with an entry field identifier that matches the Web page field identifier. Therefore, Larsen does not teach or suggest the features of claims 4, 16 and 24.

With regard to dependent claim 5, which is representative of claims 17 and 25 with regard to similarly recited subject matter, Larsen does not teach sending the Web page identifier to the directory server from the background application running on the client device or receiving the user input and Web page field identifier from the directory server in response to sending the Web page identifier from the background application on the client device. The Final Office Action alleges that Larsen teaches these features at column 3, lines 64-67 and column 4, lines 3-25, which reads as follows:

The information is then transferred over the network to the Form Parser (**13**), which is a network server. The Form Parser (**13**) submits the request for the form and inputted data to the Form Database (**3**).

(Column 3, lines 64-67, Larsen)

> The Form Builder (**14**) is a computer processor that gathers all the previously stored data associated with the user's request from the relational database and separates PDF data from Non PDF data. The Non PDF data is stored in a copy of the completed form at the Completed Form (**18**) storage. The rest of the data, which is all PDF data, makes up the PDF File (**15**), and is sent to the PDF Merger (**7**).
>
> The PDF Merger (**7**), a computer that uses PDF Merger by Digital Apps, Adobe PDF fill software, or other PDF merge software well known to those skilled in the art, completes the form by merging the PDF File (**15**) with an Empty Adobe PDF File (**16**), which is the stored form template that was previously identified with the form being filled out. The filled out PDF file is then stored in the Completed Form (**18**) storage where it can be viewed at a later time. The full file consisting of the complete PDF form and the Non PDF data is also sent to the Form Send (**19**) server.
>
> The Form Send (**19**) server sends the file back over the network to an External Entity (**1**) that has need to the information submitted by the Web User (**10**). The External Entity (**1**) processes the information and stores the information for later retrieval in its Data Store (**11**).

(Column 4, lines 3-25, Larsen)

In the first section, Larsen teaches a Form Parser that submits the request for a form and input data to a Form Database. In the second section, Larsen teaches a Form Builder that retrieves stored data from a relational database and separates PDF data from Non PDF data. The PDF data is sent to the PDF Merger, which merges the PDF data with an empty PDF file. The filled out PDF file is then stored in the Completed Form storage along with the Non PDF data to be sent to the Form Send server. The Form Send server sends the full file back to the External Entity (Company), which processes the information submitted by the user and stores the information for later retrieval. However, neither of the above sections teaches or suggests sending the Web page identifier to the directory server from the background application running on the client device. As stated in the above arguments, Larsen does not teach or suggest any background application running on the client device, nor does Larsen teach a background application that sends a Web page identifier to the directory server.

As discussed in the arguments presented for claim 1, with the background application of the presently claimed invention that runs on the client device, a user thinks that he or she is visiting a Web page in a normal fashion, however, specific functions are performed in the background to retrieve user input and Web page field identifier and populate the Web page without the user's awareness. Larsen does not teach such a feature as recited in claim 5. Larsen only teaches that the user has to request a form that is stored with pre-filled fields from the user input. The system of

Larsen does not perform specific functions, such as sending a Web page identifier to the directory server in the background without user's request. Since Larsen does not teach sending the Web page identifier to the directory server from the background application running on the client device, Larsen would not teach receiving the user input and Web page field identifier from the directory server in response to sending the Web page identifier from the background application running on the client device. Thus, Appellant respectfully submits that Larsen does not teach the specific features of claims 5, 17 and 25.

With regard to dependent claim 6, which is representative of claims 18 and 26 with regard to similarly recited subject matter, Larsen does not teach inserting user input into a field of a Web page corresponding to the Web page field identifier. As discussed in the arguments presented above for claims 5, 17 and 25, Larsen does not teach receiving user input and a Web page field identifier from the directory server in response to sending the Web page identifier from the background application running on the client device. Therefore, Larsen also fails to teach inserting data into a field of the Web page that corresponds to a Web page field identifier when read in combination with claims 5, 17 and 25.

With regard to claim 7, which is representative of claims 27 and 33 with regard to similarly recited subject matter, Larsen does not teach that the user input and the Web page field identifier are stored in a Web page entry of the directory server identified by a user identifier and a Web page identifier. The Final Office Action alleges that Larsen teaches these features at columns 5-15, where Larsen teaches a client_id, which is considered as a user identifier, a tag_id and form_id, which are all sorts of Web page identifiers, and that the Web page field identifier are stored in a Web page entry (the table) of the directory server (database) identified by a user identifier and a Web page identifier (user ID and Web page ID).

However, Appellant respectfully disagrees. At column 11, Larsen teaches two tables that are stored in the database: a form_specific_data_ex table and a form_tags table. The form_specific_data_ex table includes a reference of form_data_id in the form_data table at column 9 and a reference of applicant_id in the applicant table at column 6, which includes a client_id. However, the form_data table and the applicant table are two separate tables that reside in the database. They are not residing in a single Web page entry stored in a directory server. Thus, using Larsen's system, in order to retrieve the user input and Web page field identifier from the database,

separate queries have to be sent to the two separate tables. This is contrary to the presently claimed invention, which stores the user input and Web page field identifier, a user identifier, and a Web page identifier all in a single Web page entry of the directory server. Thus, the presently claimed invention has an advantage over Larsen in that upon a retrieval request sent by a user, the user input and Web page field identifier may be directly retrieved without having to send separate queries to the database. Therefore, Larsen does not teach the features of claims 7, 27 and 33.

With regard to claim 9, which is representative of claim 35 with regard to similarly recited subject matter, Larsen does not teach that the Web page field identifier is a HyperText Mark-up Language tag. The Final Office Action alleges that Larsen teaches these features at column 18, lines 42-44, where Larsen teaches the uniquely identified stored form templates are in HTML format. However, Larsen does not teach the data fields within the uniquely identified stored form templates are HTML tags, as recited in claim 9. As known to a person of ordinary skill in the art, a HTML page may include a variety of tag types, for example, a Java Server Page (JSP) tag. Nowhere in the reference does Larsen teach or suggest that the data fields within the uniquely identified stored form templates are HTML tags. Larsen only teaches that the form templates are in HTML format, but mentions nothing about the data fields within the form templates. Therefore, Larsen does not teach the features of claims 9 and 35.

With regard to dependent claim 11, Larsen does not teach that the method is implemented using plug-in application to a Web browser. The Final Office Action alleges that Larsen teaches this feature at column 4, lines 1-20, which teaches that a Form Merger which uses PDF merge by Digital Apps, Adobe PDF fill software, or other PDF merge software well known in the art to complete the form by merging the PDF File with an Empty Adobe PDF File. The Final Office Action further alleges that since the Adobe Reader is used to implement Larsen's teaching, Larsen teaches a plug-in application to a Web browser. However, the PDF Merger that uses the Adobe software resides in a different computer (**17**) from the Web user's (**12**). Therefore, the Adobe software used is not a plug-in application to a Web browser that is running on the Web user's device. In addition, there is no teaching or suggestion in Larsen that mentions a plug-in to a Web browser. Nowhere in the reference does Larsen even mention that the Adobe software is a plug-in to a Web browser running on the Web user's device. Therefore, Larsen does not teach that the method is implemented using plug-in application to a Web browser.

With regard to dependent claim 32, Larsen does not teach receiving, from the background application running on the client device, a Web page identifier identifying a Web page or sending the user input and Web page field identifier to the background application running on the client device from the directory server. As discussed in the arguments presented above for claims 5, 17 and 25, Larsen does not teach any background application that is running on the client device, let alone receiving from the background application a Web page identifier identifying Web page or sending the user input and web page field identifier to the background application. At column 3, Larsen only teaches that a Web user may request a form over the network and enters information on his or her display. However, Larsen does not teach that a background application that runs in parallel with the display and sends a Web page identifier to a directory server or receives user input and a Web page field identifier from the directory server. Therefore, Larsen does not teach the features of claim 32.

Thus, in addition to being dependent on independent claims 1, 13, 21, 29 and 37 the dependent claims 2-7, 9, 11, 14-18, 22-27, 30-33, 35, 36 and 38-40are also allowable by virtue of their specific recited features. Accordingly, the rejections of claims 2-7, 9, 11, 14-18, 22-27, 30-33, 35, 36 and 38-40 should be withdrawn.

II. 35 U.S.C. 103(a), Alleged Obviousness, Claims 8, 19, 28 and 34

The Final Office Action rejects claims 8, 19, 28 and 34 under 35 U.S.C. 103(a) as allegedly being unpatentable over Larsen et al.(U.S. Patent No. 6,088,700) in view of Tomko (U.S. Patent No. 5,790,668). This rejection is respectfully traversed.

With regard to dependent claim 8, the Final Office Action states:

> Referring to claim 8, Larsen has taught where in the user input is stored in the directory server (see previous paragraphs.)
>
> Larsen has not explicitly taught the user input is encrypted before being stored.
>
> However, Tomko has taught the input data is encrypted before being stored at the selected address (Col 7 lines 53-55.)
>
> It would have been obvious to a person with ordinary skill in the art at the time the invention was made to modify the teaching of Larsen such that to have the user input encrypted before being stored
>
> A person with ordinary skill in the art would have been motivated to make the modification to Larsen because having the input being encrypted would provide the security to the input (see title of Tomko.)

Final Office Action dated November 20, 2003, page 6.

As described above, Larsen does not teach sending an instruction to store user input and a Web page field identifier in a directory server from a background application running on a client device or sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier. Tomko also does not teach these features. Tomko only teaches a method for secure handling of data, comprising the steps of: acquiring a database of personal identifiers and data by encrypting the data with an encryption key and associating the encrypted data with the personal identifier in the database; comparing a personal identifier of a given individual with the database and if there is a match, obtaining encrypted data associated with the matching personal identifier; obtaining a decryption key for the encrypted data by obtaining an encrypted version of decryption key from storage and performing decryption on the decryption key utilizing the personal identifier; and performing a decryption operation on the encrypted data with the key obtained (Column 1, lines 35-54). Nowhere in the reference does Tomko teach or suggest the features of claims 1, 13, 21 and 37.

In addition, Tomko does not teach that the user input is encrypted before being stored in the directory server, as recited in claims 8, 19, 28 and 34. The Final Office Action alleges that Tomko teaches these features at column 7, lines 53-55, which reads as follows:

> However, once the profile has been input, it is encrypted with a third public key, P_{k3}, before being stored at the selected address.

In the above section, Tomko teaches encrypting profile data that is input by an operator via the profile input device with a public key prior to storing the profile data at the selected address of the database. Each profile is associated with a user fingerprint entry in the database. Thus, the profile input data is secured data that is encrypted with a public key that is stored with a user's fingerprint entry in a database. This is contrary to the presently claimed invention, when read in combination with claim 1, which encrypts a user input to a Web page via a Web browser prior to storing the user input in the directory server. The profile input data of Tomko is user input to a fingerprint profile entry of a database, not user input to a Web page. In addition, Tomko's profile data is inputted via a profile input device, not a Web browser. Therefore, it would not have been obvious to a person of ordinary skill in the art to modify Larsen's teaching to include Tomko in order to have input data encrypted before storage, because Tomko's profile input data to a fingerprint entry of a database is non-analogous to a user input to a Web page via a Web browser.

The profile input data, according to Tomko, only corresponds to a user's fingerprint entry in a

database, it does not has anything to do with a user input that is entered to a Web page and stored in a directory server. Tomko is not concerned with encrypting user input to a Web page prior to storing it in a directory server. Tomko is only interested in securing handling of profile data that is stored in an entry of a database that includes a user's fingerprint. Therefore, a person of ordinary skill in the art would not take encrypting the profile input data to store in a fingerprint entry of a database and apply it to encrypt user input to a Web page via a Web browser prior to storing the user input to a directory server, without the previous disclosure of the Appellant.

In addition, the Final Office Action alleges that a person of ordinary skill in the art would have been motivated to make the modification to Larsen because having the input being encrypted would provide security to the input. However, there is no teaching or suggestion in either Larsen or Tomko to encrypt user input to a Web page via a Web browser prior to storing the user input in the directory server. Larsen is only concerned with providing a tool to allow user to complete multiple forms for multiple companies or government agencies. Larsen does not mention anything about encrypting the user input prior to transferring over the network to the Form Parser. Tomko also does not teach or suggest encrypting user input to a Web page via a Web browser prior to storing the user input in the directory server. Tomko only teaches encrypting profile input data that corresponds to a user's fingerprint entry in a database via a profile input device. Tomko is not concerned with encrypting user input to a Web page via a Web browser. Therefore, the motivation is not supported in either of Larsen or Tomko. One of ordinary skill in the art, being presented with only Larsen and Tomko and not having any prior knowledge of Appellant's claimed invention, would not have found any reason to combine the references and modify them in the particular manner that would be necessary to arrive at encrypting user input to a Web page via a Web browser prior to storing the user input to a directory server.

Furthermore, even if a person of an ordinary skill in the art is to combine Larsen and Tomko's teaching, the result would not reach the presently claimed invention. The result of combination would be encrypting user input to a Web form prior to storing the user input to an entry that corresponds to the user's fingerprint in a database. The result would not be encrypting a user input to a Web page via a Web browser prior to storing the user input in a directory server. Therefore, a person of ordinary skill in the art would not have been motivated to modify Larsen to include Tomko's teaching simply because the combination would still not teach or suggest the presently claimed invention.

Thus, in view of the above, Appellant respectfully submits that neither Larsen nor Tomko, either alone or in combination, teach or suggest the specific features of dependent claim 8. Since dependent claims 19, 28 and 34 are rejected on the same rationale as claim 8, neither Larsen nor Tomko teach or suggest the specific features of claims 19, 28 and 34. Therefore, Appellant respectfully requests withdrawal of the rejection of dependent claims 8, 19, 28 and 34 under 35 U.S.C. § 103(a).

III. 35 U.S.C. 103(a), Alleged Obviousness, Claims 10 and 36

The Final Office Action rejects claims 10 and 36 under 35 U.S.C. 103(a) as being allegedly unpatentable over Larsen et al. (U.S. Patent No. 6,088,700) in view of Call (U.S. Patent No. 6,154,739). This rejection is respectfully traversed.

With regard to dependent claim 10, the Final Office Action states:

> Referring to claim 10, Larsen has not taught the directory server in claim 1, however, Larsen has not explicitly taught wherein the directory server is an LDAP server.
>
> However, Call has taught that a LDAP server may be advantageously employed to store "entries"... (Col 20 lines 27-Col 21 lines 4; which show that LDAP server could be also viewed as a directory server.)
>
> It would have been obvious to a person with ordinary skill in the art at the time the invention was made to modify the teaching of Larsen such that to have the directory server to be the LDAP server, because LDAP server is already well known in the art to be a directory server as being developed at the University of Michigan and later further developed by Netscape Communication Corp. provides both access and update capabilities. (Col 20, lines 30-34.)
>
> A person with ordinary skill in the art would have been motivated to make the modification to Larsen because LDAP is not a newly invented server, which is already known in the art.

Final Office Action dated November 20, 2003, page 7.

As described above, neither Larsen nor Tomko teach or suggest sending an instruction to store user input and Web page field identifier in a directory server from a background application running on a client device or sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier. Call also does not teach these features. Call teaches a method and apparatus for disseminating, over the Internet, product information produced and maintained by product manufacturers using existing universal product codes (bar codes) as access keys.

A cross-referencing resource, according to Call, may take the form of an independent HTTP

server, an LDAP directory server, or the existing Internet Domain Name Service (DNS), receives Internet request messages containing all or part of a universal product code and returns the Internet address at which information about the identified product, or the manufacturer of that product, may be obtained. By using preferred Web data storage formats which conform to XML, XLS, XLink, Xpointer and RDF specifications, product information may be seamlessly integrated with information from other sources. A "web register" module can be employed to provide an internal interface between a shared sales Internet server and an otherwise conventional inventory control system, and operates in conjunction with the cross-referencing server to provide detailed product information to Internet shoppers who may purchase goods from existing stores via the Internet (Abstract).

Thus, Call only teaches a cross referencing server that receives messages for a particular product, which includes corresponding universal product code. The cross referencing server then returns the Internet address of the particular manufacturer's server which then makes the desired product information available to the user. Call does not teach that the cross referencing server sends or receives user input and Web page field identifier to or from a background application running on a client device. Call also does not teach storing user input and Web page field identifier in a directory server.

The Final Office Action alleges that Call teaches the features of claims 10 and 36 at column 20, line 27 to column 21, line 4, which reads as follows:

> The Internet LDAP protocol may be used to advantage to implement the product code translation process. This protocol, developed at the University of Michigan and later further developed by Netscape Communication Corp. provides both access and update capabilities, allowing directory information to be created and managed as well as queried. LDAP is an open Internet standard, produced by the Internet Engineering Task Force (IETF), the same body responsible for creating TCP/IP, the Internet domain name service (DNS), and the hypertext transport protocol (HTTP). The LDAP protocol is defined in RFCs 1777 and 1778 and informational documentation is further provided in RFC 1823. The use of LDAP to provide directory lookup services via the Internet is further detailed in the literature. See, for example, *Implementing LDAP* by Mark Wilcox (Wrox Press-1999) and *LDAP – Programming Directory Enabled Applications with Lightweight Directory Access Protocol* by Tim Howes and Mark Smith (Mcmillan Technology Series – 1997). Operational LDAP server software may be purchased from a variety of sources, and includes the "Netscape Directory Server" marketed by the Netscape Communications Corporation.
>
> An LDAP server may be advantageously employed to store "entries", each of which is uniquely identified by a distinguished name (DN) which may take the form of the company code portion of the universal product code, creating a "flat namespace" in a single tree level

structure, with the remainder of the entry including a string specifying the URL of the server resource from which information about products assigned that company code may be found. In one arrangement, an online merchant's server may send a request to a remote directory server using the LDAP protocol to obtain the URL at which information about a specific product is available. Next, the merchant's server could again use the LDAP protocol to fetch information about a specific product designated by the remainder of the universal product code from a second LDAP server at the URL specified by the first server, the second LDAP server being operated by the product manufacturer to store the URL at which data describing particular products is stored. The actual product data may advantageously be stored as XML "documents" as discussed later. (emphasis added)

In the above section, Call teaches using a LDAP server to store entries, each of the entries contains a company code portion of the universal product code sent by the user and a URL from which information about products assigned that company code may be found. However, when read in combination of claims 1 and 29, Call does not teach the LDAP server as a directory server that stores user input entered by a user to a Web page via a Web browser and corresponding Web page field identifier. While Call teaches LDAP server as a directory server that stores entries, the entries does not include any user input or Web page field identifier.

To the contrary, each of the entries in Call includes a company code portion of a universal product code of a product selected by a user and a URL that identifies a source of information for the product. Neither portion of the entry as taught by Call includes user input that is entered to a Web page via a Web browser or corresponding Web page field identifier. An example of user input to a Web page, as described on page 13 of the presently claimed specification, is a social security number that is associated with a Web page field identifier <socsec>. Neither the product code of the universal product code nor the URL as taught by Call represent such user input to a Web page or corresponding Web page field identifier. Therefore, Call does not teach the same LDAP directory server that is recited in claims 10 and 36.

The Final Office Action further alleges that it would have been obvious to a person with ordinary skill in the art at the time the invention was made to modify the teachings of Larsen to have the directory server to be the LDAP server, because the LDAP server is already well known in the art to be a directory server as being developed at the University of Michigan and later further developed by Netscape Communication Corp., which provides both access and update capabilities. Appellant respectfully disagrees. While LDAP protocol is well known in the art to be used as a directory server, the LDAP server, as taught by Call, is not used in a manner that

would be necessary to arrive at an LDAP server of the presently claimed invention, which stores user input and Web page field identifier. Therefore, a person of ordinary skill in the art would not have been led to modify Larsen to include the LDAP server of Call to store user input and Web page field identifiers, simply because there is no teaching or suggestion in either Larsen or Call of an LDAP directory server that stores user input and Web page field identifier in an LDAP directory server.

Furthermore, the Final Office Action alleges that a person of ordinary skill in the art would have been motivated to make the modification to Larsen because LDAP is not a newly invented server, which is already known in the art. However, while a LDAP server is well known as a directory server that stores entries, there is no suggestion or teaching in either Larsen or Call that supports the motivation that the entries of an LDAP directory server include user input and Web page field identifier. Larsen is only concerned with providing a tool that allows a Web user to complete multiple Web forms for multiple companies. The Web user transfers information over the network to a Form Parser, which sends the request and inputted data to a Form Parser. Larsen does not suggest that a need exists for an LDAP server to store the inputted data. Call is only concerned with using LDAP directory server to store a company code portion of a universal product code of a product selected by a user and a URL that contains information for the product assigned with the company code. Call does not suggest storing user input from a Web user to the LDAP server. Therefore, a person of ordinary skill in the art would not have been motivated to modify or combine Larsen and Call, because the motivation of storing user input and Web page field identifier in an LDAP directory server is simply not supported in either of the references.

In view of the above, Appellant respectfully submits that neither Larsen nor Call, either alone or in combination, teach or suggest the specific features of dependent claim 10. Since dependent claim 36 is rejected on the same rationale as claim 10, neither Larsen nor Call teach or suggest the specific features of claim 36. Therefore, Appellant respectfully requests withdrawal of the rejection of dependent claims 10 and 36 under 35 U.S.C. § 103(a).

IV. Summary

In summary, Larsen, Tomko and Call fail to teach or suggest the specific features of the present invention. There is nothing in Larsen, Tomko or Call that teaches or suggests sending an

instruction to store a user input and a Web page field identifier in a directory server from a background application running on a client device or sending a request from a background application running on the client device to retrieve user input and the corresponding Web page field identifier from the directory server. In addition, there is nothing in Larsen, Tomko or Call that teaches or suggests encrypting the user input to a Web page via a Web browser prior to storing the user input in a directory server or an LDAP directory server that stores user input to a Web page.

V. Conclusion

In view of the above, Appellant respectfully submit that claims 1-11, 13-19 and 21-40 define over the prior art of record, Larsen, Tomko and Call. Appellant therefore respectfully requests the Board of Patent Appeals and Interferences to overturn the rejection of claims 1-11, 13-19 and 21-40 under 35 U.S.C. 103(a).

Respectfully submitted,

Stephen J. Walder, Jr.
Reg. No. 41,534
Yee & Associates, P.C.
PO Box 802333
Dallas, TX 75380
(972) 367-2001

SJW/im

APPENDIX OF CLAIMS

The text of the claims involved in the appeal are:

1. A method for maintaining state information for Web pages, comprising:

receiving user input to a Web page via a Web browser at a client device;

sending an instruction to store user input and a Web page field identifier in a directory server from a background application running on the client device;

storing the user input and a corresponding Web page identifier in a the directory server; and

in response to receiving a user request, via the Web browser, for the Web page, sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier, wherein the user input and corresponding Web page field identifier are retrieved from the directory server.

2. The method of claim 1, wherein the user input and Web page field identifier are specific to a particular Web page.

3. The method of claim 1, wherein the user input and Web page field identifier are common to a plurality of Web pages.

4. The method of claim 1, further comprising:

matching the Web page field identifier to an entry field identifier located in the Web page; and

inserting the user input into a field associated with the entry field identifier.

5. The method of claim 1, further comprising:

receiving a Web page retrieval request having a Web page identifier identifying the Web page from a Web browser running on the client device;

sending the Web page identifier to the directory server from the background application running on the client device; and

receiving the user input and Web page field identifier from the directory server in response to sending the Web page identifier from the background application running on the client device.

6. The method of claim 5, further comprising inserting the user input into a field of the Web page corresponding to the Web page field identifier.

7. The method of claim 1, wherein the user input and the Web page field identifier are stored in a Web page entry of the directory server identified by a user identifier and a Web page identifier.

8. The method of claim 1, wherein the user input is encrypted before being stored in the directory server.

9. The method of claim 1, wherein the Web page field identifier is a HyperText Mark-up Language tag.

10. The method of claim 1, wherein the directory server is an LDAP server.

11. The method of claim 1, wherein the method is implemented using a plug-in application to a Web browser.

13. A computer program product in a computer readable medium for maintaining state information for Web pages, comprising:

first instructions for receiving user input to a Web page via a Web browser at a client device;

second instructions for sending an instruction to store user input and a Web page field identifier in a directory server from a background application running on the client device;

third instructions for storing the user input and a corresponding Web page identifier in a the directory server; and

fourth instructions for sending a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier, in response to receiving a user request, via the Web browser, for the Web page, wherein the user input and corresponding Web page field identifier are retrieved from the directory server.

14. The computer program product of claim 13, wherein the user input and Web page field identifier are specific to a particular Web page.

15. The computer program product of claim 13, wherein the user input and Web page field identifier are common to a plurality of Web pages.

16. The computer program product of claim 13, further comprising:

fifth instructions for matching the Web page field identifier to an entry field identifier located in the Web page; and

sixth instructions for inserting the user input into a field associated with the entry field identifier.

17. The computer program product of claim 13, further comprising:

fifth instructions for receiving a Web page retrieval request having a Web page identifier identifying the Web page from a Web browser running on the client device;

sixth instructions for sending the Web page identifier to the directory server from the background application running on the client device; and

seventh instructions for receiving the user input and Web page field identifier from the directory server in response to sending the Web page identifier from the background application running on the client device.

18. The computer program product of claim 17, further comprising eighth instructions for inserting the user input into a field of the Web page corresponding to the Web page field identifier.

19. The computer program product of claim 13, further comprising third instructions for encrypting the user input before storing the user input in the directory server.

21. An apparatus for maintaining state information for Web pages, comprising:

a processor;

an input device coupled to the processor; and

a network interface coupled to the processor and to a network, wherein the processor receives user input to a Web page via a Web browser at a client device, sends an instruction from a background application running on the client device to a directory server, via the network interface, to store the user input and a corresponding Web page field identifier in the directory server, and sends a request from the background application running on the client device to the directory server to retrieve the user input and corresponding Web page field identifier, in response to receiving a user request via the Web browser, for the Web page, wherein the user input and corresponding Web page field identifier are retrieved from the directory server.

22. The apparatus of claim 21, wherein the user input and Web page field identifier are specific to a particular Web page.

23. The apparatus of claim 21, wherein the user input and Web page field identifier are common to a plurality of Web pages.

24. The apparatus of claim 21, wherein the processor matches the Web page field identifier to an entry field identifier located in the Web page and inserts the user input into a field associated with the entry field identifier.

25. The apparatus of claim 21, wherein the processor receives a Web page retrieval request having a Web page identifier identifying the Web page from a Web browser running on the client device via the input device, sends the Web page identifier from the background application running on the client device to the directory server via the network interface, and receives the user input and Web page field identifier from the directory server in response to sending the Web page identifier from the background application running on the client device via the network interface.

26. The apparatus of claim 25, wherein the processor inserts the user input into a field of the Web page corresponding to the Web page field identifier.

27. The apparatus of claim 21, wherein the user input and the Web page field identifier are stored in a Web page entry of the directory server identified by a user identifier and a Web page identifier.

28. The apparatus of claim 21, wherein the processor encrypts the user input before the user input is stored in the directory server.

29. A method, in a directory server, for maintaining state information for Web pages, comprising:

receiving user input to a Web page from a background application running on a client device; and

storing the user input and a corresponding Web page field identifier received from the background application running on the client device in the directory server, wherein the user input and corresponding Web page field identifier are downloaded from the directory server to the client device in response to a user request for a Web page.

30. The method of claim 29, wherein the user input and Web page field identifier are specific to a particular Web page.

31. The method of claim 29, wherein the user input and Web page field identifier are common to a plurality of Web pages.

32. The method of claim 29, further comprising:

receiving, from the background application running on the client device, a Web page identifier identifying the Web page;

retrieving the user input and Web page field identifier in response to receiving the Web page identifier; and

sending the user input and Web page field identifier to the background application running on the client device from the directory server.

33. The method of claim 29, wherein the user input and the Web page field identifier are stored as a Web page entry identified by a user identifier and a Web page identifier.

34. The method of claim 29, wherein the user input is encrypted before being stored.

35. The method of claim 29, wherein the Web page field identifier is a HyperText Mark-up Language tag.

36. The method of claim 29, wherein the directory server is an LDAP server.

37. A method in a data processing system for maintaining state information for a Web page, the method comprising:

receiving a user input to the Web page and an identifier associated with the user input via a Web browser at a client;

sending an instruction from a background application running on the client to store user input to the Web page and the identifier associated with the user input; and

storing the user input to the Web page and the identifier associated with the user input in a server; and

responsive to a subsequent request via the Web browser for the Web page, sending a request from the background application running on the client to the server to retrieve the user input and the identifier from the server.

38. The method of claim 37, wherein the server is a directory server.

39. The method of claim 37, wherein the identifier is a tag from the Web page.

40. The method of claim 37, further comprising:

placing the user input retrieved from the server in the Web page using the identifier.